arcadian

NUMBER 7 VOLUME ONE June 15,1979

CHECKERS in the last issue had one typo; in line 1220 where part of the line read: @((S+R)+2)=3; and it should have read: @((S+R)÷2)=3;
An error in line 8 had too many zeros at 3ØØØØ.
I've had many comments on this program, all pleased with the effect and operation (once the glitch was cleared up).

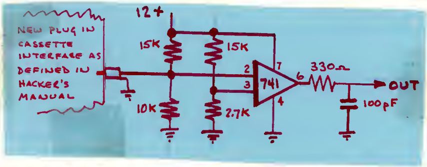
PROOFREADING of listings is getting better, and hopefully I'm not going to have any more problems as I am now sending a copy of the ready-to-print listing to the author for his last minute check.

GIANT LETTERS program on p.45 can be 'clarified' by replacing line 105 with the statement X=Ø; GOSUB C . The zero will stop the machine's printing after it finishes the AN of ARCADIAN. What is happening is - the machine has been set into a printing mode and it keeps on going until it hits an internal halt. The X=Ø sets such a halt thru the POKE function, where you want it.

CONNECTOR for the 50-pin expansion port is a standard printed circuit device, with 25 double-sided pins on 0.1 inch centers. I understand that the APPLE II uses this, as well as the SHUGART disc drive. There are some of these on the surplus market here, 3M No. 3415, and I can supply these at $2.50 ppd, including a 3" bit of flat cable that you could solder to.

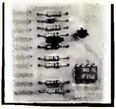
KEYBOARD project in work utilizes a CHERRY brand (?) keyboard wired in parallel with the keypad with some buffers so that either can be used. All letters and characters are where they belong, while the shift key allows the generation of the various WORDS. The Keyboard is the one advertised by JAMES ELECTRONICS, 1021 Howard Ave., San Carlos CA 94070, at $29.95. "63 Key Uncoded Keyborad".

PRINTER project has been made to work ,using a Type 43 Teletype machine and a kit for an interface available from ELECTRONIC SYSTEMS, P.O. Box 21638, San Jose, CA 95151, their part #232A at $7. plus postage. I saw an ad of theirs in KILOBAUD, April, p.172. The schematic of this kit is included for you scratch-builders:


NEW PLUG IN CASSETTE INTERFACE AS DEFINED IN HACKER'S MANUAL
12+
15K
15K
10K
2.7K
2
3
7
741
4
6
330Ω
OUT
100pF

RS-232/ TTL INTERFACE

• Converts TTL to RS-232, and converts RS-232 to TTL • Two separate circuits • Requires -12 and +12 volts • All connections go to a 10 pin gold plated edge connector • Board only $4.50 Part No. 232, with parts $7.00 Part No. 232A



COLOR STANDARDS question came up, and I find that Bally would make an excellent color generator, as apparently their color output is very well controlled. I wonder if one of the TV-technical types can compare colors with a standard color generator output and tell us which color numbers give you the standard shades necessary for color TV work.

BANGMAN game program included this month is by Ernie Sams,248 S. Forest Street, Bellingham WA 98225. It has a novel twist over the old hangman game and is quite clever. It has a good scheme for entering characters without their appearing on the screen, and a search routine that can locate and account for multi-usage of a letter. I am also including Ernie's sheet of documentation that will be of help to a lot of us.

```
Line #  Statement(s)
     3  .BANGMAN
     5  .BY E.SAMS (c)3-3-79
     7  .
     9  NT=2
    10  CLEAR;PRINT"ONE PERSON
        KEYS IN A WORD TO 1ϕ LET-
        TERS;ANOTHER TRIES TO
        GUESS IT WITH NO
    20  PRINT"MORE THAN 9 WRONG
        GUESSES USING KN & TR(1)
    30  PRINT;PRINT"PRESS GO
    50  I=KP;IF I=13 CLEAR
   100  E=ϕ;Q=ϕ
   200  PRINT"ENTER WORD, THEN
        PRESS GO.
   300  FOR A=ϕ TO 9
   310  CY=31
   320  PRINT"LETTER #",#3,A+1
   330  BOX-55,23,5ϕ,8,2
   400  K=KP
   410  IF K=13 GOTO 79ϕ
   420  IF K>9ϕ PRINT"INVALID";
        GOTO 31ϕ
   430  IF K<65 PRINT"INVALID";
        GOTO 31ϕ
   600  @(A)=K;@(A+1ϕ)=K
   700  NEXT A
   790  CLEAR
   800  FOR B = ϕ TO A-1
   900  Y=-32;X=-8ϕ+(B×8)+3
  1010  BOX X,Y,7,3,1;NEXT B
  1100  CY=-24;CX=6;PRINT"USED LET
        TERS";CY=-32;CX=42;
        PRINT "BELOW:
  1200  FOR M=ϕTO25;@(M+20)=
        M+65;NEXT M
  1300  FOR D=ϕTO25
  1400  CY=31;CX=-8ϕ
  1410  PRINT"GUESS #",#3,D+1
  1420  BOX-33,19,94,16,2
```

```
Line #  Statement(s)
  1430  G=(KN(1)+132)÷1ϕ+65
  1440  CY=23;CX=-74;TV=G
  1450  IF TR(1)=ϕ GOTO 143ϕ
  1540  IF@(G-45)=-1 PRINT"HAS
        BEEN USED"; GOTO 14ϕϕ
  1550  @(G-45)=-1;CY=-40;CX=
        -78+((G-65)×6);TV=G
  1600  FOR C=ϕ TO A-1
  1700  CY=-25
  1800  IF G=@(C) CX=-79+(C×8)+3;
        TV=G; Q=1; @(C)=-1
  1900  NEXT C
  2000  IF Q≠1 GOSUB 96ϕϕ+(E×1ϕ);
        E=E+1;IF E=9 GOTO 9ϕϕϕ
  2050  Q=ϕ
  2100  FOR F=ϕ TO A-1
  2110  IF @(F)#-1 GOTO 22ϕϕ
  2120  NEXT F
  2130  CY=17;CX=-5ϕ
  2140  PRINT"CONGRATULATIONS
  2150  CY=ϕ;CX=-8ϕ;PRINT"PRESS
        GO";GOTO 5ϕ
  2200        NEXT D
  9000  CX=-8ϕ;CY=-9
  9010  PRINT"SORRY ABOUT THIS
  9020  PRINT"THE WORD WAS
  9030  FOR H=ϕ TO A-1
  9035  CX=-79+((H×8)+3)
  9040  TV=@(H+1ϕ);NEXT H
  9060  BOX-16,ϕ,1ϕ,1ϕ,1;BOX-8,7,
        27,5,1;BOX-19,11,1,2,1;BOX
        4,11,1,2,1;BOX-9,4,1,4,1
  9110  BOX-6,3,1,5,1;BOX-8,ϕ,6,1,
        1;BOX-18,7,3,3,2;BOX-17,7,
        2,1,1;BOX-14,7,3,3,2
  9160  BOX-14,7,1,1,1;BOX-11,7,1,
        3,2;BOX-9,6,2,1,2;BOX-7,8,
        3,1,2;BOX-7,7,1,2,2
  9200  &(9)=222;&(ϕ)=87;&(1)=87;
        &(2)=8ϕ;&(3)=8ϕ
```

```
Line #  Statement(s)
  9225  FOR U=ϕ TO 5
  9230  &(23)=255; &(21)=255
  9240  &(23)=ϕ; &(21)=31
  9260  FOR V=5TO2ϕ;&(19)=V;NEXT V
  9270  &(21)=ϕ; &(19)=ϕ
  9300  X=RND(1ϕ)+55
  9310  Y=RND(12)
  9320  BOX X,Y,1,1,2
  9340  NEXT U
  9350  FC=8ϕ; &(1ϕ)=ϕ
  9360  BOX 6ϕ,1ϕ,38,6ϕ,2
  9370  BOX-8,4,3ϕ,18,2
  9400  FOR Y=ϕTO176;&(1ϕ)=Y;NEXTY
  9410  &(9)=5ϕ; FC=ϕ
  9500  CY=-5ϕ;CY=16;PRINT"PRESS
        GO"; GOTO 5ϕ
  9600  BOX 6ϕ,27,1ϕ,14,1;BOX 58,
        28,2,1,2;BOX 62,28,2,1,2;
        BOX 6ϕ,24,6,1,2;RETURN
  9610  BOX 6ϕ,18,4,5,1;RETURN
  9620  BOX 6ϕ,6,16,2ϕ,1;RETURN
  9630  BOX 5ϕ,14,4,3,1;BOX 49,6,
        3,2ϕ,1;RETURN
  9640  BOX 7ϕ,14,4,3,1;BOX 7ϕ,6,
        3,2ϕ,1;RETURN
  9650  BOX 57,-12,3,15,1;RETURN
  9660  BOX 6ϕ,-12,3,15,1;RETURN
  9670  BOX 55,-20,8,2,1;RETURN
  9680  BOX 67,-20,8,2,1;RETURN

 NOTE:  ELIMINATE AS MANY SPACES
        AS POSSIBLE(EXCEPT THOSE
        INSIDE QUOTES).IF YOU RUN
        SHORT,CUT LINES 3+30
```

BANGMAN DOCUMENTATION:

```
  10 -  200  Initialize and instructions.
 300 -  700  Ask for and accept up to ten valid letters.
        600  Stores the word in two locations:
               1.  To keep track of the letters correctly guessed.
               2.  To print the word if not guessed in nine tries.
 800 - 1010  Set up blanks for the word.
       1200  Store each letter of the alphabet for future use so the
             same letter is not used twice.
       1300  Initiate guessing loop.  Allows 26 guesses.
1430 - 1450  Allows one to guess a letter by turning knob #1.
       1540  If storage position is set to -1 the letter has been used.
       1550  Sets value of storage position to -1 and prints letter at
             the bottom of the screen.
1600 - 1900  Loop through the storage positions in 600 (1).  If a match
       1800  is found print the letter in the appropriate location(s)
             on the blanks established in 800-1010 and change the
             storage position value to -1.  Set flag "Q" to 1.
       2000  If the flag "Q" is not equal to 1 then the chosen letter
             did not match a letter in the word so go to subroutine
             9600 plus counter E times 10 and print that portion of the
             man.  Increment the E counter.  If there have been nine
             wrong guesses default to 9000 to "bang" part of bangman.
       2050  Otherwise flag "Q" equals 0.
2100 - 2200  Loop through storage positions in 600 (1).  If all positions
             are -1 then the word has been guessed.  Print "congratula-
             tions" and press go to start a new game.
9000 - 9020  The word was not guessed within the nine wrong guesses
             allowed.  The man was completely built, so - -
9030 - 9040  Print out the word (from 600 (2)) on the blanks.
9060 - 9160  Draw a gun with the word COLT on it.
       9200  Change the screen to a border format.
9225 - 9340  Put six random shots in the body of the man.  Use sound
             effects.  9230-9240 is the shot.  9260 is the ricochet.
       9350  Blank out screen
9360 - 9370  Blank out man
       9400  Uncover screen from top to bottom with man gone.
       9410  Restore screen to full screen format.
       9500  Press go to start a new game.
SUBROUTINES
       9600  Prints head, eyes, mouth.
       9610  Prints neck.
       9620  Prints body.
       9630  Prints right arm.
       9640  Prints left arm.
       9650  Prints right leg.
       9660  Prints left leg.
       9670  Prints right foot.
       9680  Prints left foot.
```

RANDOM ART has been expanded with some added sound and shape enhancements by Dave Stocker. Add the following to last month's program:

```
      15 &(Ø) =7; &(1)=7; &(9)=84; NT=Ø,&(21)=14; &(22)=255
      85 B=FC+RND(31)x8+4; &(2)=B; &(3)=B
     125 &(19)=X; &(18)=Y
     135 IF &(23)=1 RUN
         Press GO to set new parameters.     (try 5,13)
```

TUTORIALS by Steve Walters (556 Langfield, Northville, ME 48167) and Dave Iback (19553 Dartmouth) follow. The first provides some very interesting basic data on character size and how to make them appear exactly where you want them. The second provides additional comments on the IF-type statements discussed on p. 41.

CHARACTER SIZE AND PRINT LOCATION

Every character is 5 pixels wide, with a one-pixel space added to the right of the character to provide a one-pixel separation between characters. Thus, the effective width of a character is 6 pixels.

Every character is 7 pixels high, with a one-pixel space added beneath it to provide a one-pixel separation between lines of print. Thus, the effective height of a character is 8 pixels.

The cursor is displayed as a 6-pixel wide by 8-pixel high box, the effective size of a character. When a program is running, the cursor is not visible. When a program ends, the cursor is displayed wherever it happens to be, preceeded by the line entry indicator > .

If a PRINT command is not ended with a comma, the computer will leave a full space (6 pixels by 8 pixels) following the last character in the PRINT statement, and shift to the next print line. This end-of-statement space will appear as a white box against a black background, and will blank out anything located beneath it.

If a PRINT command is ended with a comma, no space is added beyond the one-pixel space to the right of the last character, and the cursor remains at that location until another PRINT command is given.

A character prints centered on its CY location, but not centered on its CX location. Given CX and CY as the print location of a character, the horizontal center of the character is CX-1 and the vertical center of the character is CY. The left edge of the character is located on CX-3 and the right edge of the character is located on CX+1. The top and bottom of the character are located on CY+3 and CY-3, respectively.

Since the left edge of the screen display area is on CX=-80, and the left edge of a character is on CX-3, a character will print at CX=-77 even if the program specifies CX=-78, -79 or -80 prior to the PRINT command. However, this behavior is not duplicated at the right side of the screen. If CX is specified at +78, the right edge of the character will be on CX=+79 (CX+1, and the right limit of the screen display area); but if CX is specified at +79, the character will print beyond the CX=+79 limit. The cursor will shift lines in the process, and if a comma follows the PRINT command, the cursor will shift to CX=-77 on the same CY line.

SCREEN DISPLAY AREA RELATED TO CHARACTER SIZE

The edge limits of the screen display area (CX=-80 to +79; CY=43 to -44) are functionally related to the character print size. The normal top line of print (without a CY value being specified) is CY=40, and thus the top of the characters on that line are at CY=43. There are 11 normal print lines, located at CY=40, 32, 24, 16, 8, 0, -8, -16, -24, -32 and -40. The bottom line (CY=-40) results in the bottom of the characters on that line being at CY=-43 and the one-pixel space below them being at CY=-44.

Similarly, the screen display area width is 160 pixels. This would allow 26 characters (26x6=156 pixels) with 4 pixels to spare. However, the first line of an entry starts with the line entry indicator ❭ as the first character, and so only 25 characters and spaces can be entered. When 25 additional characters have been entered, the one-pixel space to the right of the last character is at CX=75: an additional character cannot fit in the remaining 4 pixels so the cursor shifts to the next line. On second and subsequent lines of an entry, 26 characters can be entered per line.

When a program is being LISTed, the address of each line entry starts 2 characters to the right of CX=-80, at CX=-68. Thus, the first line including address will contain 24 characters and spaces; and the second and subsequent lines will contain 26 characters and spaces.

BOX COMMANDS RELATED TO PRINT LOCATION

The dimensions of a box which will outline a printed statement, or blank it out, can be readily determined from the CX and CY values of the PRINT command.

Framing a printed statement or blanking it out with a reverse box. The smallest box which can frame a character is a reverse box:

```
BOX CX-1, CY, 7,9,3
```

The above box is a rectangle which is 7 pixels wide and 9 pixels high superimposed over the 5 pixel by 7 pixel character.

A more esthetically pleasing reverse box frame is obtained from a square box:

```
BOX CX-1,CY,9,9,3
```

A similar formula can be used to blank out a character:

```
BOX CX-1,CY5,7,2
```

Note that the reverse box must be produced after the PRINT command for the statement being framed or blanked out has been executed.

Framing a printed statement with an outline box. The outline box X is more useful than the reverse frame box because the outline box can be produced before the PRINT command is executed, and thus the character can be printed and reprinted inside the outline box as desired during a program.

The smallest outline box which can frame a character is:

```
BOX CX-1,CY,9,11,1
BOX CX-1,CY,7, 9,2
```

Again, a square outline box is more esthetically pleasing:

```
BOX CX-1,CY,11,11,1
BOX CX-1,CY, 9, 9,2
```

Outline box for a statement of more than one character. Given CX and CY as the PRINT start location for a row of "n" characters (n=1 or more), with a comma following the PRINT instruction, the outline box which will frame the row of "n" characters can be determined from the following:

```
BOX CX-1+3(n-1),CY,11+6(n-1),11,1
BOX CX-1+3(n-1),CY, 9+6(n-1), 9,2
```

Note: a reverse box could also frame the row of characters using either of the above BOX commands with a 3 as the ending code.

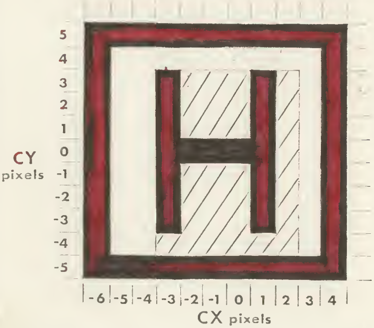
If a comma is not desired to follow the PRINT instruction, then a larger outline box is required because of the 6-pixel space added after the last character:

```
BOX CX-1+3(n-1),CY,21+6(n-1),11,1
BOX CX-1+3(n-1),CY,19+6(n-1), 9,2
```

Illustration of an outline box and a printed character inside it:

```
BOX CX-1,CY,11,11,1
BOX CX-1,CY, 9, 9,1
[illegible]
```

▨ Area [illegible] "H" is printed



TUTORIAL 2 is a follow-up on last issue's comments on IF statements, again by Dave and Steve.

The Bally BASIC interprets IF statements in terms of Boolean algebra concepts. In simple terms, each condition in an IF statement is assigned a value of one (+1) if it is true(i.e., if it is met) or a value of zero (Ø) if it is false (not met).

The program then executes the IF statement if the resulting Boolean value of the IF statement is greater than zero, or proceeds to the next program line if the value is zero.

Each condition in an IF statement must be placed in parentheses if there are two or more conditions, but it is optional if there is only one condition. Thus, simple IF statements may be programmed in either of two ways:

```
1000 IF A=4 GOTO 2000                  (uses 12 bytes)
1000 IF (A=4) GOTO 2000                (uses 14 bytes)
```

(if A=4, a value of 1 is given to the IF statement and the program branches to 2000)

```
1000 IF A=4 IF B=5 GOTO 2000           (uses 16 bytes)
1000 IF (A=4)+(B=5)=2 GOTO 2000        (uses 22 bytes)
```

(if A=4, a value of 1 is given to the IF statement, and if B=5, a value of 1 is also given, so that if both conditions are true the resulting value is 2 and the program branches to 2000; if only one of the two, or neither, is true, the program does not branch)

Obviously, in both of the above cases, using the parentheses costs more in memory bytes and is thus not efficient. However, in more complicated IF statement conditions, parentheses result in less memory bytes being used; and in some cases the saving is substantial.

For example, consider a condition where A=4 or B=5 or both (i.e.,A=4 and/or B=5);

```
1000 IF A=4 GOTO 2000
1010 IF B=5 GOTO 2000                  (uses 24 bytes)

1000 IF (A=4)+(B=5) GOTO 2000          (uses 20 bytes)
```

Even more memory savings would be involved if the condition was any one or more of three variable values.

The following list illustrates other applications of special IF statements in the most byte-efficient form, based on the above principals:

Special IF statements:

```
     If A=1 and/or B=2        1000  IF (A=1) + (B=2) GOTO 2000
     If A=1 and B=2           1000  IF A=1 IF B=2 GOTO 2000
     If A=1 or B=2 but not both       1000 IF (A=1) - (B=2) GOTO 2000
     If A ≠ ∅ (i.e., if ABS (A) ∅)    1000 IF A GOTO 2000
         This is also useful for hand control responses:
         1000 IF TR(1) GOTO 2000      will branch if trigger is pulled
         1000 IF JX(1) GOTO 2000      will branch if joystick moved sideways
         1000 IF JY(1) GOTO 2000      will branch if joystick moved back & forth
     If neither A not B equal zero    1000 IF A IF B GOTO 2000
     If A=1 and B=2 and C=3           1000 IF A=1 IF B=2 IF C=3 GOTO 2000
     If A=1 and/or B=2 and/or C=3     1000 IF (A=1) + (B=2) + (C=3) GOTO 2000
     If A=1 or B=2 or C=3 but not more
       than one of these              1000 IF(A=1) + (B=2) + (C=3) =1 GOTO 2000
     If any two of the above three
       conditions are to be met       1000 IF(A=1) + (B=2) + (C=3) =2 GOTO 2000
```

TUTORIAL 3 is another follow-up, this time by Jean Taillefer.

The >= and =< symbols will work in an IF statement. They stand for "equal to or greater than" and "less than or equal to", respectively. The symbols are sometimes shown as ≥ or ≤.

The statement IF A>=B GOTO 100 is equivalent to

```
IF A> B GOTO 100;IF A=B GOTO 100     or
IF(A>B) + (A=B) GOTO 100
```

A statement that is equivalent to the 'reverse sign' function on a calculator can be accomplished with the following: -(A)

If A is positive, the statement is evaluated as -(+A), for a negative number.

If A is negative, the statement is evaluated as -(-A), for a positive number.

This statement can be used with a variety of other statements, such as:

```
PRINT -(A)
B=-(A)
IF -(A) = 1 GOTO 100
```

This is different from the ABS command which always gives a positive number.

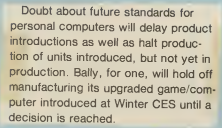
DEALER in the St. Louis area is RTS Electronics/Gametronics, 356 Brookes Dr. 731-2309 who report that they have all cartridges in stock, do mail order, and hold Football tournaments for players on Tuesdays,6-9pm and Saturdays, noon to 4. They mention that players are bringing their own hand controllers, which makes me think of the slot car operators...

DEALER in Indiana is ABC Hobbycraft 2155 E.Morgan Ave,Evansville 47711 (477-9661) who are open Sundays, and who act as the local hangout for Bally enthusiasts.

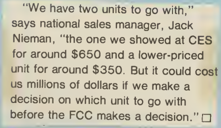
GAMES 2007 PINBALL and 2006 SPACE INVADERS are scheduled for July release. There were a few words on these on p. 23.

STATEMENT When I started this paper late last year, I indicated in my subscription form that I would print material as it became available, but at least bi-monthly in this fiscal year. Well, I have been doing better than bi-monthly because material has been coming in at a much higher rate than I expected, so I have been publishing more or less monthly. This is a hobby output so publication is bound to be somewhat haphazard. Post-printing operations are starting to get unwieldy and I may have to get professional help in the folding/stapling area. Right now it looks like there will be at least 2 more issues to November, so you'll be getting at least a 50% 'profit'.

PROGRAMMABLE KEYBOARD? The latest delay hinges on a Texas Instruments petition now before the FCC. They have asked for a change in the procedures by which computer/parts are approved for use. Apparently RCA tried this two years ago but never pushed very hard. The current procedure is for a computer manufacturer to have his whole system approved as meeting RFI, TVI, etc., limits, while the proposal is to have only the RF modulator inspected/approved. This has thrown the manufacturers into a tizzy as they feel that if the rules or standards are changed in the middle of the stream they'll have design and manufacturing costs to contend with. Bally, who haven't gone into production yet, feels they have to wait and see which way the FCC goes to avoid the 50-50 chance of being in error when the decision is made. For a company that thrives on gambling, those are unacceptable odds. Here is an excerpt from Consumer Electronics monthly 5/79 p. 31:



Doubt about future standards for personal computers will delay product introductions as well as halt production of units introduced, but not yet in production. Bally, for one, will hold off manufacturing its upgraded game/computer introduced at Winter CES until a decision is reached.



"We have two units to go with," says national sales manager, Jack Nieman, "the one we showed at CES for around $650 and a lower-priced unit for around $350. But it could cost us millions of dollars if we make a decision on which unit to go with before the FCC makes a decision." □

My discussions with Bally indicate that they expect a corporate discussion/decision around January, with delivery months later than that. Needless to say, the whole situation is disappointing and frustrating. Fortunately, we have some people working on ways to 'make our own', and we can see a little glimmer of light, as reported earlier this issue. Subscribers who are working on a keyboard,memory addition, or any other "goodie" are urged to write me so that I can build up a team to get the needed hardware built and available for us all.

ARCADIAN
Robert Fabris, stapler
3626 Morrie Dr.
San José,CA 95127

* See note on Page 2. BANGMAN by Ernie Sams

7

OK

s, using KN & TR #1

```
  10 NT=2;CLEAR;PRINT "ONE PERSON KEYS IN A WORD TO 10 LETTERS; THE
     OTHER USES KN & TR #1 TO GUESS--"
  20 PRINT "ONLY 9 WRONG GUESSES ALLOWED."
  40 PRINT;PRINT "PRESS GO"
  50 I=KP;IF I=13 CLEAR
 100 E=0; Q=0
 200 PRINT "ENTER WORD, THEN PRESS GO."
 300 FOR A=0 TO 9
 310 CY=31
 320 PRINT "LETTER #",#3,A+1
 330 BOX -55,23,50,8,2
 400 K=KP
 410 IF K=13 GOTO 790
 420 IF K>90 PRINT " INVALID";GOTO 310
 430 IF K<65 PRINT " INVALID";GOTO 310
 600 @(A)=K; @(A+10)=K
 700 NEXT A
 790 CLEAR
 800 FOR B=0 TO A-1
 900 Y=-32;X=-80+(Bx8)+3
1010 BOX X,Y,7,3,1; NEXT B
1100 CY=-24;CX=6;PRINT "USED LETTERS";CY=-32;CX=42;PRINT "BELOW:"
1200 FOR M=0 TO 25; @(M+20)=M+65; NEXT M
1300 FOR D=0 TO 25
1400 CY=31; CX=-80
1410 PRINT "GUESS #",#3,D+1
1420 BOX -33,19,94,16,2
1430 G=(KN(1)+132)÷ 10+65
1440 CY=23;CX=-74;TV=G
1450 IF TR(1)=0 GOTO 1430
~~1520 IF G>90 PRINT " INVALID"; GOTO 1400~~
~~1530 IF G<65 PRINT " INVALID"; GOTO 1400~~
1540 IF @(G-45)=-1 PRINT " HAS BEEN USED"; GOTO 1400
1550 @(G-45)=-1; CY=-40; CX=-78+((G-65)x6); TV=G
1600 FOR C=0 TO A-1
1700 CY=-25
1800 IF G=@(C) CX=-79+(Cx8)+3; TV=G: Q=1; @(C)=-1
1900 NEXT C
2000 IF Q#1 GOSUB 9600+(Ex10); E=E+1; IF E=9 GOTO 9000
2050 Q=0
2100 FOR F=0 TO A-1
2110 IF @(F)#-1 GOTO 2200
2120 NEXT F
2130 CY=17; CX=-50
2140 PRINT "CONGRATULATIONS"
2150 CY=0; CX=-80; PRINT "PRESS GO"; GOTO 50
2200 NEXT D
9000 CX=-80; CY=-9
9010 PRINT "SORRY ABOUT THIS"
9020 PRINT "THE WORD WAS"
9030 FOR H=0 TO A-1
9035 CX=-79+((Hx8)+3)
9040 TV#@(H+10); NEXT H
9060 BOX -16,0,10,10,1; BOX -8,7,27,5,1; BOX -19,11,1,2,1;
     BOX 4,11,1,2,1; BOX -9,4,1,4,1
```

```
9110 BOX -6,3,1,5,1; BOX -8,0,6,1,1; BOX -18,7,3,3,2;
     BOX -17,7,2,1,1; BOX -14,7,3,3,2
9160 BOX -14,7,1,1,1; BOX -11,7,1,3,2; BOX -9,6,2,1,2;
     BOX -7,8,3,1,2; BOX -7,7,1,2,2
9200 &(9)=222; &(0)=87; &(1)=87; &(2)=80; &(3)=80
9225 FOR U=0 TO 5
9230 &(23)=255; &(21)=255
9240 &(23)=0; &(21)=31

9260 FOR V=5 TO 20; &(19)=V; NEXT V
9270 &(21)=0; &(19)=0
9300 X=RND(10)+55
9310 Y=RND(12)
9320 BOX X,Y,1,1,2
9340 NEXT U
9350 FC=80; &(10)=0
9360 BOX 60,10,38,60,2
9370 BOX -8,4,30,18,2
9400 FOR Y=0 TO 176; &(10)=Y; NEXT Y
9410 &(9)=50; FC=0
9500 CX=-50; CY=16; PRINT "PRESS GO":GOTO 50
9600 BOX 60,27,10,14,1; BOX 58,28,2,1,2; BOX 62,28,2,1,2;
     BOX 60,24,6,1,2; RETURN
9610 BOX 60,18,4,5,1; RETURN
9620 BOX 60,6,16,20,1; RETURN
9630 BOX 50,14,4,3,1; BOX 49,6,3,20,1; RETURN
9640 BOX 70,14,4,3,1; BOX 70,6,3,20,1; RETURN
9650 BOX 57,-12,3,15,1; RETURN
9660 BOX 64,-12,3,15,1; RETURN
9670 BOX 55,-20,8,2,1; RETURN
9680 BOX 67,-20,8,2,1; RETURN
```

NOTE: Because of the length of this program any spaces outside of quotation marks should not be keyed into the program (unless the computer automatically puts them in). Spaces were typed in to make the program more readable.

There must be at least 90 unused bytes remaining.

BANGMAN DOCUMENTATION:

```
  10 -  200  Initialize and instructions.
 300 -  700  Ask for and accept up to ten valid letters.
        600  Stores the word in two locations:
               1.  To keep track of the letters correctly guessed.
               2.  To print the word if not guessed in nine tries.
 800 - 1010  Set up blanks for the word.
       1200  Store each letter of the alphabet for future use so the
             same letter is not used twice.
       1300  Initiate guessing loop.  Allows 26 guesses.
1430 - 1450  Allows one to guess a letter by turning knob #1.
       1540  If storage position is set to -1 the letter has been used.
       1550  Sets value of storage position to -1 and prints letter at
             the bottom of the screen.
1600 - 1900  Loop through the storage positions in 600 (1).  If a match
       1800  is found print the letter in the appropriate location(s)
             on the blanks established in 800-1010 and change the
             storage position value to -1.  Set flag "Q" to 1.
       2000  If the flag "Q" is not equal to 1 then the chosen letter
             did not match a letter in the word so go to subroutine
             9600 plus counter E times 10 and print that portion of the
             man.  Increment the E counter.  If there have been nine
             wrong guesses default to 9000 to "bang" part of bangman.
       2050  Otherwise flag "Q" equals 0.
2100 - 2200  Loop through storage positions in 600 (1).  If all positions
             are -1 then the word has been guessed.  Print "congratula-
             tions" and press go to start a new game.
9000 - 9020  The word was not guessed within the nine wrong guesses
             allowed.  The man was completely built, so - -
9030 - 9040  Print out the word (from 600 (2)) on the blanks.
9060 - 9160  Draw a gun with the word COLT on it.
       9200  Change the screen to a border format.
9225 - 9340  Put six random shots in the body of the man.  Use sound
             effects.  9230-9240 is the shot.  9260 is the ricochet.
       9350  Blank out screen
9360 - 9370  Blank out man
       9400  Uncover screen from top to bottom with man gone.
       9410  Restore screen to full screen format.
       9500  Press go to start a new game.
SUBROUTINES
       9600  Prints head, eyes, mouth.
       9610  Prints neck.
       9620  Prints body.
       9630  Prints right arm.
       9640  Prints left arm.
       9650  Prints right leg.
       9660  Prints left leg.
       9670  Prints right foot.
       9680  Prints left foot.
```

```
Line #    Statement(s)                          Comments
     3    .BANGMAN
     5    .BY E.SAMS (C) 3-3-79
     7    .
     9    NT=2
    10    CLEAR;PRINT"ONE PERSON
          KEYS IN A WORD TO 1Ø LET-
          TERS;ANOTHER TRIES TO
          GUESS IT WITH NO
    20    PRINT"MORE THAN 9 WRONG
          GUESSES USING KN & TR(1)
    30    PRINT;PRINT"PRESS GO
    50    I=KP;IF I=13 CLEAR
   100    E=Ø;Q=Ø
   200    PRINT"ENTER WORD, THEN
          PRESS GO.
   300    FOR A=Ø TO 9
   310    CY=31
   320    PRINT"LETTER #",#3,A+1
   330    BOX-55,23,5Ø,8,2
   400    K=KP
   410    IF K=13 GOTO 79Ø
   420    IF K>9Ø PRINT"INVALID";
          GOTO 31Ø
   430    IF K<65 PRINT"INVALID";
          GOTO 31Ø
   600    @(A)=K;@(A+1Ø)=K
   700    NEXT A
   790    CLEAR
   800    FOR B = Ø TO A-1
   900    Y=-32;X=-8Ø+(Bx8)+3
  1010    BOX X,Y,7,3,1;NEXT B
  1100    CY=-24;CX=6;PRINT"USED LET
          TERS";CY=-32;CX=42;
          PRINT "BELOW:
  1200    FOR M=ØTO25;@(M+20)=
          M+65;NEXT M
  1300    FOR D=ØTO25
  1400    CY=31;CX=-8Ø
  1410    PRINT"GUESS #",#3,D+1
  1420    BOX-33,19,94,16,2
```

USE OF SHADED AREA IS FOR 2ND OR MORE LINES OF MULTI-LINE STATEMENTS

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT, THIS IS DONE BY THE UNIT

Line # | Statement(s) | Comments

```
1430 G=(KN(1)+132)÷1Ø+65
1440 CY=23;CX=-74;TV=G
1450 IF TR(1)=Ø GOTO 143Ø
1540 IF@(G-45)=-1 PRINT"HAS
     BEEN USED"; GOTO 14ØØ
1550 @(G-45)=-1;CY=-40;CX=
     -78+((G+65)x6);TV=G
1600 FOR C=Ø TO A-1
1700 CY=-25
1800 IF G=@(C) CX=-79+(Cx8)+3;
     TV=G; Q=1; @(C)=-1
1900 NEXT C
2000 IF Q≠1 GOSUB 96ØØ+(Ex1Ø);
     E=E+1;IF E=9 GOTO 9ØØØ
2050 Q=Ø
2100 FOR F=Ø TO A-1
2110 IF @(F)#-1 GOTO 22ØØ
2120 NEXT F
2130 CY=17;CX=-5Ø
2140 PRINT"CONGRATULATIONS
2150 CY=Ø;CX=-8Ø;PRINT"PRESS
     GO";GOTO 5Ø
2200      NEXT D
9000 CX=-8Ø;CY=-9
9010 PRINT"SORRY ABOUT THIS
9020 PRINT"THE WORD WAS
9030 FOR H=Ø TO A-1
9035 CX=-79+((Hx8)+3)
9040 TV=@(H+1Ø);NEXT H
9060 BOX-16,Ø,1Ø,1Ø,1;BOX-8,7,
     27,5,1;BOX-19,11,1,2,1;BOX
     4,11,1,2,1;BOX-9,4,1,4,1
9110 BOX-6,3,1,5,1;BOX-8,Ø,6,1,
     1;BOX-18,7,3,3,2;BOX-17,7,
     2,1,1;BOX-14,7,3,3,2
9160 BOX-14,7,1,1,1;BOX-11,7,1,
     3,2;BOX-9,6,2,1,2;BOX-7,8,
     3,1,2;BOX-7,7,1,2,2
9200 &(9)=222;&(Ø)=87;&(1)=87;
     &(2)=8Ø;&(3)=8Ø
```

USE OF SHADED AREA IS FOR 2ND OR MORE LINES OF MULTI-LINE STATEMENTS

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT, THIS IS DONE BY THE UNIT

Line # Statement(s) Comments

```
9225 FOR U=Ø TO 5
9230 &(23)=255; &(21)=255
9240 &(23)=Ø; &(21)=31
9260 FOR V=5TO2Ø;&(19)=V;NEXT V
9270 &(21)=Ø; &(19)=Ø
9300 X=RND(1Ø)+55
9310 Y=RND(12)
9320 BOX X,Y,1,1,2
9340 NEXT U
9350 FC=8Ø; &(1Ø)=Ø
9360 BOX 6Ø,1Ø,38,6Ø,2
9370 BOX-8,4,3Ø,18,2
9400 FOR Y=ØTO176;&(1Ø)=Y;NEXTY
9410 &(9)=5Ø; FC=Ø
9500 CX=-5Ø;CY=16;PRINT"PRESS
     GO"; GOTO 5Ø
9600 BOX 6Ø,27,1Ø,14,1;BOX 58,
     28,2,1,2;BOX 62,28,2,1,2;
     BOX 6Ø,24,6,1,2;RETURN
9610 BOX 6Ø,18,4,5,1;RETURN
9620 BOX 6Ø,6,16,2Ø,1;RETURN
9630 BOX 5Ø,14,4,3,1;BOX 49,6,
     3,2Ø,1;RETURN
9640 BOX 7Ø,14,4,3,1;BOX 7Ø,6,
     3,2Ø,1;RETURN
9650 BOX 57,-12,3,15,1;RETURN
9660 BOX 64,-12,3,15,1;RETURN
9670 BOX 55,-20,8,2,1;RETURN
9680 BOX 67,-20,8,2,1;RETURN
```

NOTE:ELIMINATE AS MANY SPACES AS POSSIBLE(EXCEPT THOSE INSIDE QUOTES).IF YOU RUN SHORT,CUT LINES 3→30

USE OF SHADED AREA IS FOR 2ND OR MORE LINES OF MULTI-LINE STATEMENTS

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT,THIS IS DONE BY THE UNIT

Bally Arcade

May 20, 1979

Steve Walters
556 Langfield
Northville, Mi 48167

CHARACTER SIZE AND PRINT LOCATION

Every character is 5 pixels wide, with a one-pixel space added to the right of the character to provide a one-pixel separation between characters. Thus, the effective width of a character is 6 pixels.

Every character is 7 pixels high, with a one-pixel space added beneath it to provide a one-pixel separation between lines of print. Thus, the effective height of a character is 8 pixels.

The cursor is displayed as a 6-pixel wide by 8-pixel high box, the effective size of a character. When a program is running, the cursor is not visible. When a program ends, the cursor is displayed wherever it happens to be, preceeded by the line entry indicator > .

If a PRINT command is not ended with a comma, the computer will leave a full space (6pixels by 8 pixels) following the last character in the PRINT statement, and shift to the next print line. This end-of-statement space will appear as a white box against a black background, and will blank out anything located beneath it.

If a PRINT command is ended with a comma, no space is added beyond the one-pixel space to the right of the last character, and the cursor remains at that location until another PRINT command is given.

A character prints centered on its CY location, but not centered on its CX location. Given CX and CY as the print location of a character, the horizontal center of the character is CX-1 and the vertical center of the character is CY. The left edge of the character is located on CX-3 and the right edge of the character is located on CX+1. The top and bottom of the character are located on CY+3 and CY-3, respectively.

Since the left edge of the screen display area is on CX=-80, and the left edge of a character is on CX-3, a character will print at CX=-77 even if the program specifies CX=-78, -79 or -80 prior to the PRINT command. However, this behavior is not duplicated at the right side of the screen. If CX is specified at +78, the right edge of the character will be on CX=+79 (CX+1, and the right limit of the screen display area); but if CX is specified at +79, the character will print beyond the CX=+79 limit. The cursor will shift lines in the process, and if a comma follows the PRINT command, the cursor will shift to CX=-77 on the same CY line.

SCREEN DISPLAY AREA RELATED TO CHARACTER SIZE

The edge limits of the screen display area (CX=-80 to +79; CY=43 to -44) are functionally related to the character print size. The normal top line of print (without a CY value being specified) is CY=40, and thus the top of the characters on that line are at CY=43. There are 11 normal print lines, located at CY=40, 32, 24, 16, 8, 0, -8, -16, -24, -32 and -40. The bottom line (CY=-40) results in the bottom of the characters on that line being at CY=-43 and the one-pixel space below them being at CY=-44.

Similarly, the screen display area width is 160 pixels. This would allow 26 characters (26x6=156 pixels) with 4 pixels to spare. However, the first line of an entry starts with the line entry indicator > as the first character, and so only 25 characters and spaces can be entered. When 25 additional characters have been entered, the one-pixel space to the right of the last character is at CX=75: an additional character cannot fit in the remaining 4 pixels so the cursor shifts to the next line. On second and subsequent lines of an entry, 26 characters can be entered per line.

When a program is being LISTed, the address of each line entry starts 2 characters to the right of CX=-80, at CX=-68. Thus, the first line including address will contain 24 characters and spaces; and the second and subsequent lines will contain 26 characters and spaces.

BOX COMMANDS RELATED TO PRINT LOCATION

The dimensions of a box which will outline a printed statement, or blank it out, can be readily determined from the CX and CY values of the PRINT command.

Framing a printed statement or blanking it out with a reverse box. The smallest box which can frame a character is a reverse box:

```
BOX CX-1,CY,7,9,3
```

The above box is a rectangle which is 7 pixels wide and 9 pixels high superimposed over the 5 pixel by 7 pixel character.

A more esthetically pleasing reverse box frame is obtained from a *square* box:

```
BOX CX-1,CY,9,9,3
```

A similar formula can be used to blank out a character:

```
BOX CX-1,CY,5,7,2
```

Note that the reverse box must be produced *after* the PRINT command for the statement being framed or blanked out has been executed.

Framing a printed statement with an outline box. The outline box X is more useful than the reverse frame box because the outline box can be produced before the PRINT command is executed, and thus the character can be printed and reprinted inside the outline box as desired during a program.

The smallest outline box which can frame a character is:

```
BOX CX-1,CY,9,11,1
BOX CX-1,CY,7, 9,2
```

Again, a *square* outline box is more esthetically pleasing:

```
BOX CX-1,CY,11,11,1
BOX CX-1,CY, 9, 9,2
```

Outline box for a statement of more than one character. Given CX and CY as the PRINT start location for a row of "n" characters (n=1 or more), with a comma following the PRINT instruction, the outline box which will frame the row of "n" characters can be determined from the following:

```
BOX CX-1+3(n-1),CY,11+6(n-1),11,1
BOX CX-1+3(n-1),CY, 9+6(n-1), 9,2
```

Note: a reverse box could also frame the row of characters using either of the above BOX commands with a 3 as the ending code.

If a comma is not desired to follow the PRINT instruction, then a larger outline box is required because of the 6-pixel space added after the last character:

```
BOX CX-1+3(n-1),CY,21+6(n-1),11,1
BOX CX-1+3(n-1),CY,19+6(n-1), 9,2
```

Illustration of an outline box and a printed character inside it:

```
BOX CX-1,CY,11,11,1
BOX CX-1,CY, 9, 9,2
CX=0; CY=0; PRINT "H",
```

▨ Area cleared when "H", is printed

Bally Arcade

IF Statements

May 20, 1979

Dave Ibach
19553 Dartmouth
Northville, Mi

Steve Walters
556 Langfield
Northville, Mi 48167

The Bally Basis interprets IF statements in terms of Boolean algebra concepts. In simple terms, each condition in an IF statement is assigned a value of one (+1) if it is true (i.e., if it is met) or a value of zero (0) if it is false (not met).

The program then executes the IF statement if the resulting Boolean value of the IF statement is greater than zero, or proceeds to the next program line if the value is zero.

Each condition in an IF statement must be placed in parentheses if there are two or more conditions, but it is optional if there is only one condition. Thus, simple IF statements may be programed in either of two ways:

```
1000 IF A=4 GOTO 2000                  (uses 12 bytes)
1000 IF (A=4) GOTO 2000                (uses 14 bytes)
```

(if A=4, a value of 1 is given to the IF statement and the program branches to 2000)

```
1000 IF A=4 IF B=5 GOTO 2000           (uses 16 bytes)
1000 IF (A=4)+(B=5)=2 GOTO 2000        (uses 22 bytes)
```

(if A=4 a value of 1 is given to the IF statement, and if B=5 a value of 1 is also given, so that if both conditions are true the resulting value is 2 and the program branches to 2000; if only one of the two or neither is true, the program does not branch)

Obviously, in both of the above cases, using the parentheses costs more in memory bytes and is thus not efficient. However, in more complicated IF statement conditions, parentheses result in less memory bytes being used; and in some cases the savings is substantial.

For example, consider a condition where A=4 or B=5 or both (i.e., A=4 <u>and/or</u> B=5):

```
1000 IF A=4 GOTO 2000
1010 IF B=5 GOTO 2000                  (uses 24 bytes)

1000 IF (A=4)+(B=5) GOTO 2000          (uses 20 bytes)
```

Even more memory savings would be involved if the condition was any one or more of <u>three</u> variable values.

The attached list illustrates other applications of special IF statements in the most byte-efficient form, based on the above principals.

Bally Arcade Dave Ibach Steve Walters May 20, 1979

Special IF statements

IF A=1 and/or B=2	1000 IF (A=1)+(B=2) GOTO 2000
IF A=1 and B=2	1000 IF A=1 IF B=2 GOTO 2000
IF A=1 or B=2 but not both	1000 IF (A=1)-(B=2) GOTO 2000
IF A#0 (i.e., IF ABS(A) > 0)	1000 IF A GOTO 2000

This is also useful for hand control responses:

1000 IF TR(1) GOTO 2000	will branch if trigger is pulled
1000 IF JX(1) GOTO 2000	will branch if joystick is moved left or right
1000 IF JY(1) GOTO 2000	will branch if joystick is moved forward or backward

IF neither A nor B equal zero	1000 IF A IF B GOTO 2000
IF A=1 and B=2 and C=3	1000 IF A=1 IF B=2 IF C=3 GOTO 2000
IF A=1 and/or B=2 and/or C=3	1000 IF (A=1)+(B=2)+(C=3) GOTO 2000
IF A=1 or B=2 or C=3 but not more than one of these	1000 IF (A=1)+(B=2)+(C=3)=1 GOTO 2000
IF any two of three conditions are met	1000 IF (A=1)+(B=2)+(C=3)=2 GOTO 2000

Sweeney

May 7, 1979 Ansd 5/24

Dear Bob,

Sorry about the long delay, but things have been extremely busy around here. I did look at all memory-reference programs on the Bally, and it definitely seems that the load-read ability of my program is what we need. However, my effort is still pretty primitive. I haven't had time to improve it, but I have used it to develop about 300 bytes of machine-language routine residing in my memory-expander (now 2K). It proved invaluable, and I'm sure a similar program, with variable-base I/O + better control structures will be even better.

send data

I have come up with a way of making faster tapes. The machine spends most of its time waiting for the display to scroll. By cutting out the number of scrolling operations, the tape operation is sped up 2 to 4X.

Sweeney

The program to be loaded has to have predictable line numbering (say every 10, using GO+10).

Also, it is better if each line number occupies only one line. If the program statements are consistently longer than this what the "8" in 20020 (and "80" in 20010) can be reduced. Remember that the LIST M, N has to cover every statement number. There is nothing wrong, except loss of efficiency, with listing some statements more than once.

The X is replaced by the highest line number that is to be recorded; the "step 80" is for a program numbered every 10. This value + the constant in 20020 are chosen to fill the screen with 8 lines of text, followed by a "CLEAR".

The "CLEAR" (in QUOTES) in 20030 has to be the WORDS - CLEAR. The program doesn't have to be at 20000, but can be anywhere, + may or may not list itself; it is appended to the program to be taped, + then you turn on the tape recorder, "GOTO 20000".

On playback, (:Input), the CLEAR'S ARE interpreted as screen clear, + prevent the scrolling.

A ":LIST" will generally not keep up with

Sweeney

```
20000    CLEAR;:PRINT ;NT=1
20010    FOR  M = 0  TO  X   STEP  80
20020    LIST  M, 8
20030    PRINT "CLEAR"; CLEAR
20040    NEXT M
```

the tape, and the tape might be too fast unless you clear the screen before :Input.

Enclosed is a check for $2.00. Please forward a copy of the "hacker's manual." I am really looking forward to doing something with the light pen interface.

I'll be writing you again, soon.

Sincerely,

John Sweeney

P.S. I finally pried a replacement for my blown I/O chip from Bally. (Through Digitrends in Cleveland). They charged $35.00.

call direct, swap

May 22, 1979

Dear Bob,

Here are some more tidbits about Bally Basic:

— The >=, <= symbols will work in an IF statement. they stand respectively for "greater than or equal to" and "lesser than or equal to" thus the statement:

```
IF A>=B GOTO 100
```

is eqvalent to:

```
~~OR~~ 1-) IF A>B GOTO 100 ; IF A=B GOTO 100
       OR
       2-) IF (A>B) + (A=B) GOTO 100
```

— The "AND" construct for the IF statement printed in the last Arcadian will not work if both tests fail. Example:

```
IF (A=3) = (B=0) GOTO 120
```

if A=4 and B=1 the statement is evaluated as

```
IF (0) = (0) GOTO 120
```

— which is not Correct!

A statement which will be correct in all cases is:

```
IF (A=3) X (B=0)
```

This would be evaluated as

```
IF (0) x (0) GOTO 120
```

— which would equal 0

— A statement which is equivalent to the "reverse sign" function on a calculator can be accomplished with the following: —(A)

if A is positive, the statement is evaluated as —(+A) which gives a negative number

if A is negative, the statement is evaluated as —(⁻A)

over→

which gives a positive number
This statement can be used with a variety of other statements such as:

```
1-) PRINT -(A)
2-) B= -(A)
3-) IF -(A) = 1 GOTO 100
```

This is different from the ABS command in that the latter always gives a positive result.

- I am working on identifying all the notes produced by all symbols and letters on the keypad. So far, these seem to cover the range of human hearing from the very low (÷!) to the very high (xX). I have already determined the symbols and their order in ascending pitch, plus one extra octave which is higher than the x7 note, More on this later

Sincerely,

Jean Taillefer

Thomka

Bob Fabris,

We would like this in the next Arcadian. Keep up the good work. We wrote to Bally in support of the expansion module. adw7

Bob & Jeri Weber

W & W SØFTWARE SALES
6594 SWARTØUT RØAD
ALGØNAC, MICHIGAN 48001

DEAR BALLY BUFF,

WE HAVE REØRGANIZED ØUR SØFTWARE AND HAVE CØME UP WITH 3 CASSETTES WITH 5 PRØGRAMS PER TAPE AT $10.00 EACH. WE WILL NØ LØNGER TAKE ØRDERS TØ RECØRD ØN YØUR TAPES. THE SELECTIØNS WILL BE PRE-PACKAGED TØ ENABLE US TØ KEEP UP WITH THE DEMAND FØR ØRDERS. IT WILL ALSØ ALLØW US TIME TØ PRØGRAM NEW GAMES FØR FUTURE TAPES.

SØFTWARE FØR TINY BASIC, W & W TAPE #1

- ØTHELLØ-0 TØ 2 PLAYERS, KEEPS SCØRE THRØUGHØUT THE GAME.
- VAN GAM-1 PLAYER
- FLIGHT SIMULATØR-LEARN THE SKILL ØF FLYING.
- SUB SEARCH-1 PLAYER. FIND THE ENEMY SUB BEFØRE YØU ARE BLØWN UP.
- HANGMAN-YØU PUT IN WØRDS, MACHINE MIXES THEM UP FØR YØU.

SØFTWARE FØR TINY BASIC, W & W TAPE #2

- CØNCENTRATIØN-2 TØ 4 PLAYERS MATCH SKILLS.
- SPACE CHASE-2 PLAYERS.
- SLØT MACHINE-PRACTICE FØR VEGAS.
- GAME ØF LIFE-WATCH THE BIRTH, GRØWTH, AND DEATH ØF A CØLØNY ØF CELLS.
- MATH QUIZ-PRØBLEMS IN ADDITIØN, SUBTRACTIØN, DIVISIØN, AND MULTIPLICATIØN. SKILL LEVEL ADJUSTS TØ PLAYER.

SØFTWARE FØR TINY BASIC, W & W TAPE #3

- BIØRHYTHMS-SEE HØW YØU ARE GØING TØ DØ TØDAY.
- ALIEN PATRØL-KILL ALL THE ALIENS AND RETURN TØ BASE, BEFØRE YØU RUN ØUT ØF FUEL.
- TIC TAC TØE-1 PLAYER.
- MASTERMIND-GUESS CØMPUTER'S 4 DIGIT NUMBER.
- CALENDAR-ANY MØNTH, ANY YEAR.

ØRDERS WILL BE HANDLED AS QUICKLY AS PØSSIBLE, BUT YØU SHØULD ALLØW 3 WEEKS FØR DELIVERY.

THANK YØU,

Bob Weber

Bob,

We have expanded our tape selection. Could you please add it to the next issue.

W & W SØFTWARE SALES

TAPE #4 IN ØUR SERIES IS NØW AVAILABLE, AND INCLUDES THE FØLLØWING PRØGRAMS:

CYLØN RAIDERS-SHØØT DØWN THE 10 CYLØNS WITH YØUR VIPER
CHECKERS-1 PLAYER. PLAY AGAINST THE CØMPUTER.
BLØCK BUSTER-KNØCK DØWN THE BRICK WALLS.
MENSA TEST-8 SAMPLE IQ QUESTIØNS TØ TEST YØURSELF WITH.
RUSSIAN RØULETTE-SEE IF YØU CAN SURVIVE 10 PULLS ØF THE TRIGGER.

MØRE TAPES WILL BE FØRTHCØMING, AND WILL BE ANNØUNCED AS SØØN AS THEY ARE AVAILABLE.

P.S. How many Arcadians are there.

THANK YØU,

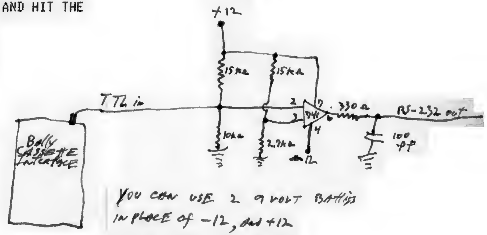
Bob & Jeri Weber

```
LIST
   5 CLEAR
  10 FOR A=1TO 20
  11 NT=RND (20)
  20 FOR I=1TO 4
  30 &(I-1)=RND (255)
  40 NEXT I
  50 &(9)=RND (255)
  60 &(10)=RND (255)
  70 FOR I=16TO 23
  80 &(I)=RND (255)
  90 NEXT I
  95 CLEAR
 100 PRINT "GOTO  GOTO  GOTO GOTO  GOTO  GOTO "
 110 NEXT A
>
```

BOB, THIS PGM WAS PRINTED WITH A TELETYPE MODEL 43 TERMINAL
THE INTERFACE IS THE ONE SOLD BY
ELECTRONIC SYSTEMS
P.O. BOX 21638
SAN JOSE, CA 95151
PART # 232A $7.00   IT IS A KIT AND COMES WITH ALL PARTS
IF SOMEONE WANTS TO BUILD ONE FROM SCRATCH HERE IS THE CKT.
this model 43 DID NOT HAVE A AUTO LINE FEED SO I HAD TO WATCH FOR THE RETURN AND HIT THE LINE FEED BUTTON.

THE PGM LISTED ABOVE IS ONE THAT SETS ALL SCREEN AND MUSIC REGISTERS TO RND.
IT COULB GO ON FOR YEARS AND NEVER REPEAT ITSELFG

JERRY L. TINDLE
8414 STAUNTON DR.
AUSTIN, TEXAS 78758



You can use 2 9 volt Batties
in place of -12, and +12
IC is 741 - op-amp

7

5, 13

T

```
                                                      new
  1 . RANDOM ART
  2 . BY E. SAMS
  3 . EXPANDED BY
  4 . DAVID STOCKER
  5 . 5/26/79
 10 X=0;Y=0;&(0)=7;&(1)=7;&(9)=84;NT=0  } IMPORTANT
 15 &(21)=14;&(22)=255
 20 INPUT "WIDTH"W
 30 INPUT "HEIGHT"H
 40 X=X+W;Y=Y+H
 50 CLEAR
 60 IF X>159W=-W;FC=RND (31)x8+4
 70 IF X<2W=-W
 80 IF Y>79H=-H;B=FC+RND (31)x8+4;&(2)=B;&(3)=B
 90 IF Y<2H=-H
                                    new
 100 X=X+W;Y=Y+H
 110 IF X<1X=1
 120 IF Y<1Y=1
 125 &(19)=X;&(18)=Y   N
 130 BOX 0,0,X,Y,3
 135 IF &(23)=1RUN   N
 140 GOTO 60
```

Press GO to set new parameters.

Stocker

8